REG NAME	REG #
$zero	0
$at	1
$v0	2
$v1	3
$a0	4
$a1	5
$a2	6
$a3	7
$t0	8
$t1	9
$t2	10
$t3	11
$t4	12
$t5	13
$t6	14
$t7	15
$s0	16
$s1	17
$s2	18
$s3	19
$s4	20
$s5	21
$s6	22
$s7	23
$t8	24
$t9	25
$k0	26
$k1	27
$gp	28
$sp	29

MNEMONIC	MEANING	TYPE	OPCODE	FUNCT
sll	Logical Shift Left	R	0x00	0x00
srl	Logical Shift Right (0-extended)	R	0x00	0x02
sra	Arithmetic Shift Right (sign-extended)	R	0x00	0x03
jr	Jump to Address in Register	R	0x00	0x08
mfhi	Move from HI Register	R	0x00	0x10
mflo	Move from LO Register	R	0x00	0x12
mult	Multiply	R	0x00	0x18
multu	Unsigned Multiply	R	0x00	0x19
div	Divide	R	0x00	0x1A
divu	Unsigned Divide	R	0x00	0x1B
add	Add	R	0x00	0x20
addu	Add Unsigned	R	0x00	0x21
sub	Subtract	R	0x00	0x22
subu	Unsigned Subtract	R	0x00	0x23
and	Bitwise AND	R	0x00	0x24
or	Bitwise OR	R	0x00	0x25
xor	Bitwise XOR (Exclusive-OR)	R	0x00	0x26
nor	Bitwise NOR (NOT-OR	[illegible]	[illegible]	[illegible]
slt	Set to 1 if Less Th	[illegible]	[illegible]	[illegible]
sltu	Set to 1 if Less Th	[illegible]	[illegible]	[illegible]
j	Jump to Address	[illegible]	[illegible]	[illegible]
jal	Jump and Link	J	0x03	NA
beq	Branch if Equal	I	0x04	NA
bne	Branch if Not Equal	I	0x05	NA
blez	Branch if Less Than or Equal to Zero	I	0x06	NA
addi	Add Immediate	I	0x08	NA
addiu	Add Unsigned Immediate	I	0x09	NA
slti	Set to 1 if Less Than Immediate	I	0x0A	NA
sltiu	Set to 1 if Less Than Unsigned Immediate	I	0x0B	NA
andi	Bitwise AND Immediate	I	0x0C	NA
ori	Bitwise OR Immediate	I	0x0D	NA
lui	Load Upper Immediate	I	0x0F	NA
mfc0	Move from Coprocessor 0	R	0x10	NA
lb	Load Byte	I	0x20	NA
lh	Load Halfword	I	0x21	NA
lw	Load Word	I	0x23	NA
lbu	Load Byte Unsigned	I	0x24	NA
lhu	Load Halfword Unsigned	I	0x25	NA
sb	Store Byte	I	0x28	NA
sh	Store Halfword	I	0x29	NA
sw	Store Word	I	0x2B	NA

MNEMONIC	MEANING	TYPE	OPCODE	FUNCT
add	Add	R	0x00	0x20
addi	Add Immediate	I	0x08	NA
addiu	Add Unsigned Immediate	I	0x09	NA
addu	Add Unsigned	R	0x00	0x21
and	Bitwise AND	R	0x00	0x24
andi	Bitwise AND Immediate	I	0x0C	NA
beq	Branch if Equal	I	0x04	NA
blez	Branch if Less Than or Equal to Zero	I	0x06	NA
bne	Branch if Not Equal	I	0x05	NA
div	Divide	R	0x00	0x1A
divu	Unsigned Divide	R	0x00	0x1B
j	Jump to Address	J	0x02	NA
jal	Jump and Link	J	0x03	NA
jr	Jump to Address in Register	R	0x00	0x08
lb	Load Byte	I	0x20	NA
lbu	Load Byte Unsigned	I	0x24	NA
lh	Load Halfword	I	0x21	NA
[illegible]	fword Unsigned	I	0x25	NA
[illegible]	er Immediate	I	0x0F	NA
[illegible]	d	I	0x23	NA
[illegible]	m Coprocessor 0	R	0x10	NA
[illegible]	Register	R	0x00	0x10
[illegible]	Register	R	0x00	0x12
[illegible]	[illegible]	R	0x00	0x18
[illegible]	iply	R	0x00	0x19
nor	Bitwise NOR (NOT-OR)	R	0x00	0x27
or	Bitwise OR	R	0x00	0x25
ori	Bitwise OR Immediate	I	0x0D	NA
sb	Store Byte	I	0x28	NA
sh	Store Halfword	I	0x29	NA
sll	Logical Shift Left	R	0x00	0x00
slt	Set to 1 if Less Than	R	0x00	0x2A
slti	Set to 1 if Less Than Immediate	I	0x0A	NA
sltiu	Set to 1 if Less Than Unsigned Immediate	I	0x0B	NA
sltu	Set to 1 if Less Than Unsigned	R	0x00	0x2B
sra	Arithmetic Shift Right (sign-extended)	R	0x00	0x03
srl	Logical Shift Right (0-extended)	R	0x00	0x02
sub	Subtract	R	0x00	0x22
subu	Unsigned Subtract	R	0x00	0x23
sw	Store Word	I	0x2B	NA
xor	Bitwise XOR (Exclusive-OR)	R	0x00	0x26

ASCII CODE						ASCII CODE				
BIN	OCT	DEC	HEX	CHARACTER		BIN	OCT	DEC	HEX	CHARACTER
010 0000	40	32	20	space		100 1110	116	78	4E	N
010 0001	41	33	21	!		100 1111	117	79	4F	O
010 0010	42	34	22	"		101 0000	120	80	50	P
010 0011	43	35	23	#		101 0001	121	81	51	Q
010 0100	44	36	24	$		101 0010	122	82	52	R
010 0101	45	37	25	%		101 0011	123	83	53	S
010 0110	46	38	26	&		101 0100	124	84	54	T
010 0111	47	39	27	'		101 0101	125	85	55	U
010 1000	50	40	28	(		101 0110	126	86	56	V
010 1001	51	41	29	)		101 0111	127	87	57	W
010 1010	52	42	2A	*		101 1000	130	88	58	X
010 1011	53	43	2B	+		101 1001	131	89	59	Y
010 1100	54	44	2C	,		101 1010	132	90	5A	Z
010 1101	55	45	2D	-		101 1011	133	91	5B	[
010 1110	56	46	2E	.		101 1100	134	92	5C	\
010 1111	57	47	2F	/		101 1101	135	93	5D	]
011 0000	60	48	30	0		101 1110	136	94	5E	^
011 0001	61	49	31	1		101 1111	137	95	5F	_
011 0010	62	50	32	2		110 0000	140	96	60	`
011 0011	63	51	33	3		110 0001	141	97	61	a
011										b
011										c
011										d
011 0111	67	55	37	7		11				
011 1000	70	56	38	8		11				
011 1001	71	57	39	9		11				
011 1010	72	58	3A	:		110 1000	150	104	68	h
011 1011	73	59	3B	;		110 1001	151	105	69	i
011 1100	74	60	3C	<		110 1010	152	106	6A	j
011 1101	75	61	3D	=		110 1011	153	107	6B	k
011 1110	76	62	3E	>		110 1100	154	108	6C	l
011 1111	77	63	3F	?		110 1101	155	109	6D	m
100 0000	100	64	40	@		110 1110	156	110	6E	n
100 0001	101	65	41	A		110 1111	157	111	6F	o
100 0010	102	66	42	B		111 0000	160	112	70	p
100 0011	103	67	43	C		111 0001	161	113	71	q
100 0100	104	68	44	D		111 0010	162	114	72	r
100 0101	105	69	45	E		111 0011	163	115	73	s
100 0110	106	70	46	F		111 0100	164	116	74	t
100 0111	107	71	47	G		111 0101	165	117	75	u
100 1000	110	72	48	H		111 0110	166	118	76	v
100 1001	111	73	49	I		111 0111	167	119	77	w
100 1010	112	74	4A	J		111 1000	170	120	78	x
100 1011	113	75	4B	K		111 1001	171	121	79	y
100 1100	114	76	4C	L		111 1010	172	122	7A	z
100 1101	115	77	4D	M						

Figure 1

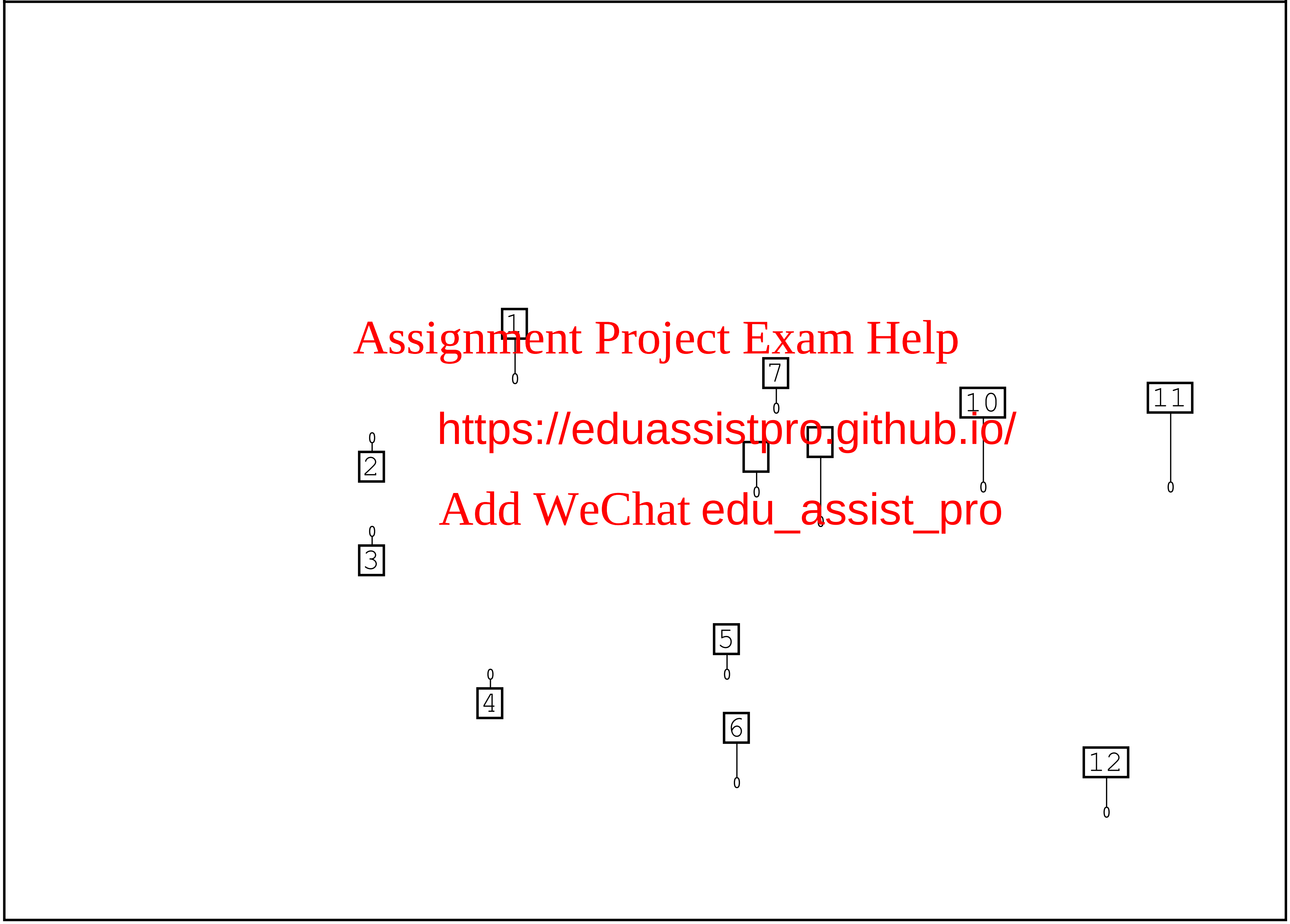


Figure 2

Register File

Reg	Initial	After Inst 1	After Inst 2	After Inst 3	After Inst 4
$t0	0x10010000	0x10010000	0x10010000	0x10010000	0x10010000
$t1	0xC0FFEEEE	0xC0FFEEEE	0xC0FFEEEE	0xC0FFEEEE	0xC0FFEEEE
$t2	0xABCDEF0	0x303FFBBB	0x303FFBBB	0x000000FB	0x000000FB
$t3	0x8000000	0x8000000	0x8000000	0x8000000	<4>

Instructions

```
SRL  $t2   $t1  <1>
<2>  $t2  ($t0)
<3>  $t2 1($t0)
ADDI $t3   $t3  0x7FFFFFFE
```

R-type fo

(shifts) instr rd

opcode	rs	rt			function
31:26	25:21	20:16	15:11	10:6	5:0

I-type format: inst rt rs immediate
inst rt immediate(rs)

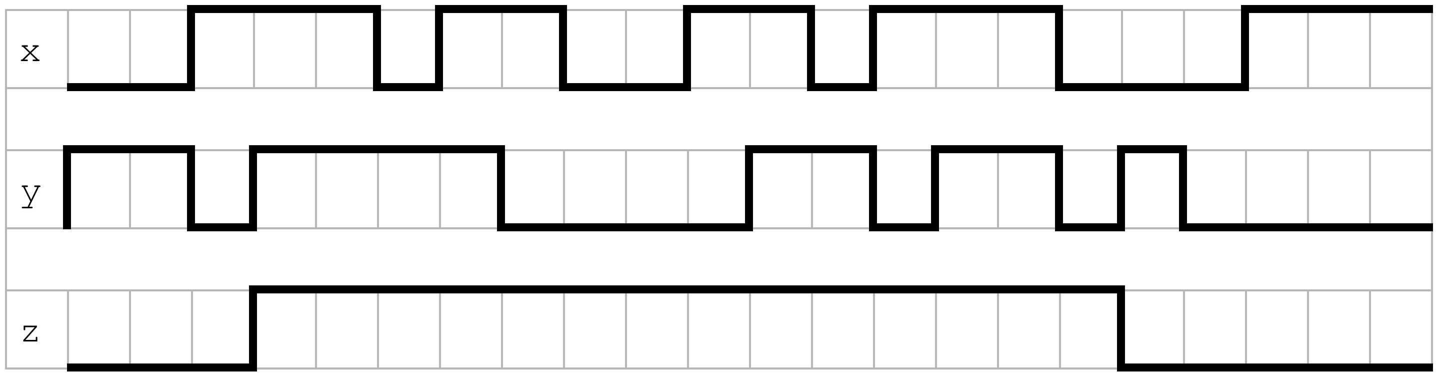
opcode	rs	rt	immediate
31:26	25:21	20:16	15:0

J-type format: j immediate

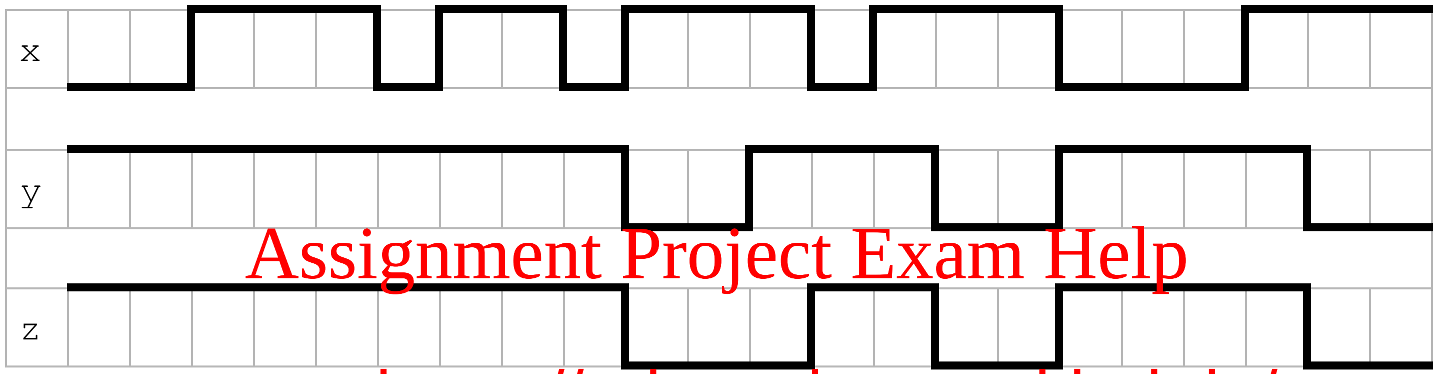
opcode	immediate
31:26	25:0

Figure 3: Timing Diagrams

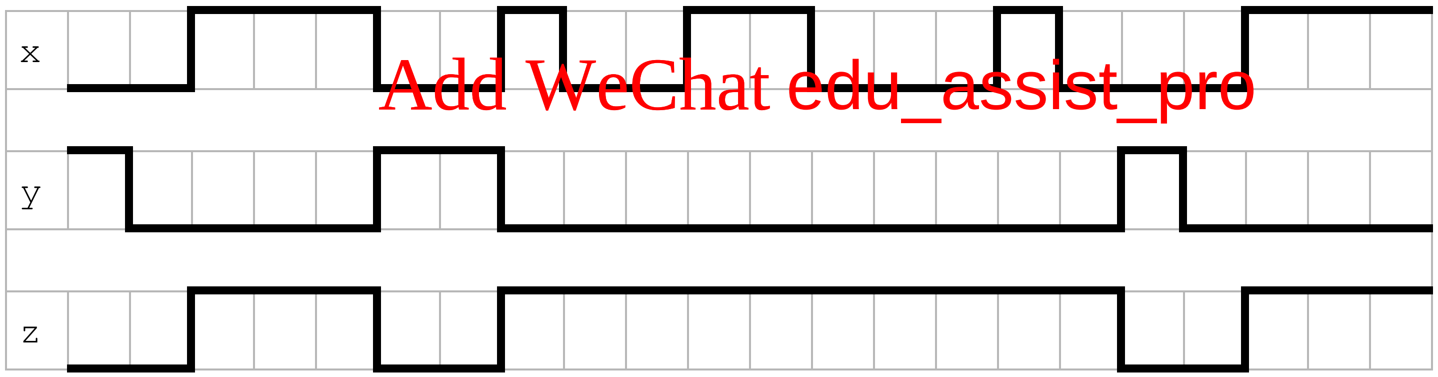
A



B



C



D